Report on the Workshop on Micro-kernels and Other Kernel Architectures
Seattle, WA, April 27–28, 1992

Peter S. Langston <psl@bellcore.com>

The first Usenix ‘‘Workshop on Micro-kernels and Other Kernel Architectures’’ was held on April twenty-seventh and twenty-eighth in sylvan Seattle Washington (a.k.a. the ‘‘Emerald City’’ for the same reason that it is a.k.a. the ‘‘Rain City’’). With three-hundred and thirty-four attendees, more than three times the anticipated number, this was not so much a workshop as an SRO conference – literally every seat in the huge meeting room was taken and some people even had to be turned away.

The rest of the title of this workshop/conference also managed to cause controversy. While some thought that the title should logically have been reduced to ‘‘Workshop on Kernel Architectures,’’ others thought that the workshop was probably aimed at comparing and contrasting existing micro-kernels and their macro-kernel counterparts and therefore should have been called ‘‘Workshop on Micro-kernels vs. Other Kernel Architectures.’’ Still others questioned the use of the term ‘‘micro-kernel’’ to describe systems that require several megabytes of memory to operate a light switch (‘‘Does this mean that MVS was really an early micro-kernel?’’). In any case, the persistent inclusion of phrases such as ‘‘and other kernels’’ betrayed the catholic intentions of the organizers.

Monday, April 27

As advertised, the first day’s sessions dealt with introductory talks on currently important micro-kernels ‘‘and other kernels.’’ After the opening remarks from Program Chair Lori Grob, five one-hour overview talks were presented.

• Robbert van Renesse (Vrije Universiteit/Cornell University) presented a ‘‘Short Overview of Amoeba’’ which was an update on a talk that has been given a few times before. His talk featured some interesting new slides including one that was the basis for Figure 1 shown here (with the addition of an NT column and minor editing by me).

• Rich Draves (Carnegie Mellon University) gave a talk on ‘‘Microkernel Operating System Architecture and Mach’’ that was characterized by one of the other paper presenters as ‘‘the most realistic Mach talk ever.’’ That evaluation may have been influenced by a section of the talk dealing with difficult decisions and things that they might do differently next time.

• Dave Presotto (AT&T Bell Laboratories) described ‘‘Plan 9, A Distributed System.’’ Aside from describing the ideas and implementation of Plan 9 (simplification and minimalism expressed as taking a few good ideas and using them to extremes), Dave also described a luxurious back-up system and provided some aphorisms: ‘‘file systems are cool,’’ ‘‘name spaces are cool,’’ and so on.

• Marc Rozier (Chorus Systèmes) gave an ‘‘Overview of the Chorus Distributed Operating System’’ that placed it both historically and technically. Much of Chorus’s terminology predates the current crop of buzzwords, making the associated paper refreshingly free of them. But not to be left behind, Marc’s talk joined the rush into the mid-90’s with a ‘‘nano-kernel’’ (≡ Real-time Executive). Marc also described ‘‘COOL’’ (Chorus Object Oriented Layer) but missed the chance to aphorize that ‘‘Chorus’s object oriented layers are cool.’’

• David Cutler (Microsoft Corporation) spoke on ‘‘Microsoft Windows NT’’ giving a broad overview ranging from Microsoft’s systems strategy and market perspective through architectural issues to time and space measurements. David made the observation that NT is hardly a microkernel and must be the ‘‘other kernel architecture’’ mentioned in the workshop title.

The next session, chaired by Dag Johansen (University of Tromsø), was on ‘‘New Architectures’’ and consisted of two papers.

• Jonathan Walpole (Oregon Graduate Institute of Science and Technology) described ‘‘Modularity and Interfaces in Micro-kernel Design and Implementation: A Case Study of Chorus on the HP-PA Risc.’’ This port of Chorus to the HP PA-RISC workstation took a year to do, uncovered common but inefficient

operating system interface architectural assumptions, and illustrated the tradeoff between micro-kernel modularity and performance. Even better, the first sentence of the paper's abstract actually answers one of the questions raised by the workshop's statement of purpose.

• Toshio Okamoto presented a paper entitled ''A Micro Kernel Architecture for Next Generation Processors'' outlining a design for an OS kernel that takes advantage of the large address space of new processors. Three design features are postulated: single virtual storage (no context switch address remapping), one-level storage (files, libraries, etc. are all parts of the single address space), and fine-grain memory protection (PTEs and two kinds of ACLs implemented by a fancy MMU). Any paper that shows the subroutine call as a new way to do message-passing gets my vote.

	Amoeba	**Mach**	**Plan 9**	**Chorus**	**NT**
Architecture	Centralized processor pool	Symmetric	Centralized processor pool	Symmetric	Symmetric
Model	Object-based	? (whatever)	File-based	Object-based	Object-based
Communication	RPC multicast	Message + RPC	Streams + file system RPCs	Message + RPC + unreliable multicast	LPC + RPC
Naming	Capabilities + directory service	Port rights + naming service	File name space (directories)	Capabilities	Unified name space per machine
Protection	Capabilities	Port rights communication based	Owner/group/other (owner can be a set of users)	Capabilities	All objects protected with ACLs
Light-weight processes	Yes, kernel-scheduled	Yes	No	Yes	Yes, kernel-scheduled
Unix support	Slow source emulation	Yes	Almost exactly Unix with library level Posix	Yes	Posix support
Distributed applications (across net)	Excellent support	OK	Not really	OK	Yes, RPC based
Multiprocessor support	Yes	Excellent	Great UMA (SMP) support	Yes	Excellent
Virtual memory	Segments	Paging	Paging	Paging	Paging
Fault tolerance	Replicated services	No explicit support	Great backups	Dynamic reconfiguration	Mirroring, striping, duplexing, & others

Figure 1

The final session of the day was the Micro-kernels Panel Session moderated [sic] by Peter Honeyman (University of Michigan). The panelists were: Dave Presotto, David Cutler, Rich Draves, Jim Lipkis (Chorus Systèmes), and Robbert van Renesse. It is not unusual for the moderator of such a panel discussion to have to calm down the panelists and act as peacemaker; that did not happen. What did happen? Well, here's what I was able to write down. I started out indicating all the places where there was general laughter or applause, but there were so many that I had to give it up. ''Floor'' is used to indicate a question or comment from ''the floor.'' For the panel members I have used initials (so only really smart people will know who said what).

P.H.: It's all lies! It's all the same bloat as a Unix kernel – so why is it ''micro''?
R.D.: You can throw away the Unix part of the bloat . . .
P.H.: But then it's not useful!
J.L.: If you want to make money it will have to have Unix.
D.C.: or DOS.

. . . [discussion of some Plan 9 port that was done in 7 days – mostly the time it took Ken Thompson to port the C compiler.]

D.C.: Have you guys got any more of those porting guys:? I'd like a couple.

P.H.: I don't think you can afford them, Dave.

[D.P. talks a lot and teases Chorus for their diagrams] . . .

P.H.: Dave, what does NT stand for, anyway? It certainly couldn't be "New Technology" . . .

D.C.: If you've ever seen the inside of DOS you'd see why NT is New Technology.

P.H.: I'd like to be the first to welcome Microsoft into the 1970's.

. . .

P.H.: Everyone but Plan 9 claims to be a virtual "porting machine." . . . [P.H. asks a confusing question of J.L. and then tries again and makes it coherent and gets a careful, coherent, and unamusing answer. This is followed by an unanswered, but much more amusing question.]

. . .[D.P. breates R.D. for Mach trying to be all things for all people and wanting to make a platform that then requires everything else (e.g. Unix emulation) to be added on.]

J.L.: The world outside this room doesn't care about minimality and cleanliness . . .

D.P.: Now you're getting to the important point – none of this really matters!

. . . [An audience member asks a question about whether any of the so-called micro-kernels can run in 8K of memory]

R.R.: Amoeba will run on an 8K machine – with the right ifdefs.

[J.L. claims that minimality was investigated as a goal (for Chorus) and found "not to be a win" so all the things that were removed were put back in.]

J.L.: Minimality itself is not much of a goal.

. . .

Floor: [The questioner starts with a long list of the problems that beset developers] . . . So what have you guys done to help me with these problems?

[A l o o o o n g silence . . .]

P.H.: Well, I thought it was a bullshit question, too.

. . . [A little later another audience member asks . . .]

A.M.: Er, my question is kinda like the preceding one . . .

P.H.: Well forget it! I don't want to talk about it anymore!

D.P.: Perhaps, if you could rephrase it as a personal insult?

. . . [Mike O'Dell asks a question from the floor about distribution involving many pieces of bread, peanut butter, and a squeegee that no one understands but J.L. (maybe). So J.L. suggests a related question (that he *can* answer) and answers it. Mike seems mollified.]

. . .

Floor: What do these systems do when faced with data rates of a terabit a second?

D.P.: Does "choke and die" mean anything? [general agreement] We're still limited by our interfaces to about 10 megabytes per second.

. . .

Floor: I have two questions. Blah, blah, . . .[I didn't write down the first question] . . . blah?

P.H.: That's a bullshit question; what's your other one?

Floor: [Unfazed] Okay; Plan 9. How do you know what something's called if everything can have its own name space?

D.P.: . . .by convention . . . [he gets onto the subject of catching all filesystem references] . . . The ability to do that, the ability to circumcise the world, . . . er, . . . to circumscribe the world is immensely powerful . . . [there follows a fairly long discussion over whether Plan 9's lack of structure is a **G**ood **T**hing – D.P.'s apparent willingness to admit the possibnility of being wrong creating something of a feeding frenzy among the other panelists.]

P.H.: Well, that's it. Goodbye.

The reception that followed was distinguished only by an unusual surfeit of blue sky outside the picture windows and an unusual deficit of beer from the excellent local micro-breweries. The appearance of six bottles of Red Hook early in the night (hotel leftovers) only served to whet appetites that could not be satisfied in the hotel . . .

Tuesday, April 28

The first session on the second day was called ‘‘New Systems’’ and was chaired by Robbert van Renesse. Four papers were presented.

• Charles Landau (MACS Lab, Inc.) talked about ‘‘The KeyKOS® Nanokernel Architecture.’’ Development of this nanokernel system began in 1975 and the system was in production use by 1983. It can run in 100 kilobytes of memory and a subset of MVS has been ported to the KeyKOS platform. Designed to favor reliability and security over performance, the system requires extraordinary measures to set capabilities at initial startup, but once set they are ‘‘persistent’’ and can be retracted only by prearrangement. This makes a development problem when a test system gets ‘‘weird;’’ even pulling the plug doesn’t fix it because it is ‘‘persistently weird.’’

• Dan Hildebrand (Quantum Software Systems) gave ‘‘An Architectural Overview of QNX.’’ This new system has only existed since 1982 and was (according to Intel Corp.) the first multiprocessing O.S. on the PC. The latest version is Posix compliant and only requires 6.8K bytes of memory for the micro-kernel, but would require nearer 100K for a minimal, a.k.a. ‘‘light-switch,’’ O.S., (big enough to be a nano-kernel, I guess). Audience questions concerned clock synchronization on the LAN and plans to port to a RISC machine (no).

• E. Douglas Jensen (Digital Equipment Corporation) spoke on ‘‘An Architectural Overview Of The Alpha Real-Time Distributed Kernel.’’ This amusing talk about the distributed thread, real-time, OS kernel joint project involving Concurrent Computer Corp., D. E. C., and the Open Software Foundation contained numerous pithy quotes – ‘‘Real Fast’’ is not ‘‘Real Time’’ – ‘‘The security guys are seriously anal retentive’’ – ‘‘There’s nothing micro about Alpha.’’ Strangely enough, in support of the last statement he mentioned that the source code was 20,000 lines of C, the same number claimed for the KeyKOS nanokernel.

• W. E. Kuhnhauser (German National Research Center for Computer Science) talked about ‘‘Performance of the BirliX Operating System.’’ While the paper characterizes BirliX as ‘‘an operating system for distributed, secure, and fault-tolerant applications’’ the speaker pointed out that it may be viewed as a ‘‘persistent object management system’’ and not a micro-kernel in any case. On the other hand, this is probably the newest of the new systems that were presented in this session.

During the break before the next session, the team that had spent the previous evening investigating Micro-breweries and Other Brewery Architectures made an appearance to give a report (the principal investigator’s colouration could only be explained as a tribute to the Emerald City). Following the break, a paper session entitled ‘‘Lessons Learned,’’ chaired by Edward Lazowska (University of Washington), presented three papers.

• Jun Nakajima (Fujitsu Laboratories Ltd.) described ‘‘Multimedia / Realtime Extensions for Mach 3.0’’ making some interesting comparisons between Mach 2.5 and Mach 3.0 in the process. He divided multimedia devices into two types – response-time-sensitive (event-driven) and response-time-insensitive (deadline-driven) and showed how extensions to include ‘‘realtime threads’’ and a ‘‘temporal paging system’’ handle them.

• Henry Massalin (Columbia University) spoke about ‘‘Reimplementing the Synthesis Kernel on the NeWS Workstation.’’ Synthesis breaks most of the rules: it is written entirely in macro assembler, the kernel includes self-modifying code, it is blindingly fast (as are the programs that run on it), it is small (a minimal kernel runs in 16K RAM and 16K ROM), and is not called a micro-, nano-, pico-, or femto-kernel. Henry played a recording of some music produced by software synthesis. He mentioned that a keyboard note generator program takes 720 micro-seconds to (1) sense a key press, (2) create a thread, (3) attach the thread to the audio output, (4) start executing the thread, and (5) produce the beginning of the sound output. Henry also described some clever solutions to cache concurrency problems encountered by machines executing self-modifying code. Quincy and his daughter Emily appeared briefly and said ‘‘qua’’ encouragingly to the audience.

• William Davenport (Digital Equipment Corporation) presented ‘‘A Model and Prototype of VMS Using the Mach 3.0 Kernel.’’ Modeling VMS took 9 months; prototyping the VMS model took another 3 months. A plea for a native mode Mach debugger was made (with agreement from the audience). After implementing 46 of the 250 VMS system services several VNS utilities were found to be runnable. Conclusions were drawn: micro-kernel technology is cool and multi-server technology is cool, but performance is probably a

casualty.

Lunch was uneventful except that we got to see Historic Pike Place Market, ate a lot of Mexican food, and drank fluorescent Mexican sodas (to the horror of the aforementioned micro-brewery test team captain).

Program chair Lori Grob was also session chair for the following session yclept ‘‘Experience and Observations I’’ comprising three papers.

• Brian Bershad (Carnegie Mellon University) decried ‘‘The Increasing Irrelevance of IPC Performance for Microkernel-Based Operating Systems’’ while new Seattle resident Rick Rashid turned the slides. Four points were advanced: IPC has gotten faster faster than other stuff; caches, not address spaces, determine performance; All data does [sic] not need to go through the kernel; all services do not need a hardware firewall. The question period was initiated with the reminder that an unwritten rule disallows the slide turner from asking questions.

• Jochen Liedtke (German National Research Center) presented a paper on ‘‘Fast Thread Management and Communication Without Continuations’’ that describes the operating system L3, argues for the relevance of IPC and concludes that (1) IPC can be implemented realy fast; (2) continuations will not support this job; and (3) availability of fast IPC changes programming behavior. Confused questions ensued.

• Jim Hamrick (Unisys Corporation) discussed ‘‘Experience with SVR4 Over Chorus’’ and stressed that the project was one of very few involving commercial product development with micro-kernels rather than academic research on them. Twenty-two people spent eighteen months bringing the project to completion. The initial requirements are met and the system is stable.

The final break of the conference passed with no noteworthy occurrences. The chair for the last session ‘‘Experience and Observations II’’ was Jim Lipkis.

• Randy Dean (Carnegie Mellon University) pointed out that his talk would be different from the paper ‘‘Data Movement in Kernelized Systems’’ in that the paper strives to describe Chorus and Mach side by side while the talk just focusses on their similarities, which include: VM central caching, an external mapper, fast & reliable IPC, and a trap redirection mechanism. He concludes that kernelized systems are here [but are they cool?] and good file system performance is possible.

• Marc Shapiro (INRIA) gave a design report entitled ‘‘Distributed Abstractions, Lightweight References’’ in which a library of useful abstractions structured as fragmented objects and protocols to support lightweight, robust, uniform, garbage collected, distributed references are proposed as amendments to current operating systems designs in place of the more ‘‘heavyweight’’ ports, pipes, and sockets.

• Robbert van Renesse presented ‘‘Reliable Multicast between Microkernels,’’ describing a re-implementation of the ISIS system designed specifically to take advantage of microkernel technology and fill in some gaps in current microkernel support (e.g. cross-network communication and failure detection). One of the goals is to make the ISIS system ‘‘FTPable’’ and examples dealt with the netnews-like ‘‘ISIS news groups.’’

• Michael Stumm (University of Toronto) gave the final paper, ‘‘Designing a Scalable Operating System for Shared Memory Multiprocessors.’’ This paper proposes a structuring technique based on clustering to solve problems of scalability in multiprocessor operating system design.

As the second day came to a close and questionnaires were handed out, attendees had a chance to look back over the two days and evaluate the workshop design. The initial overview sessions established a basis of reference for the later discussion (and disabused the attendees of any notion that the term ‘‘microkernel’’ implies something about size). The following papers were both interesting and well-presented and the panel session was ... well, interestingly presented. This timely workshop dealt with a topic that, as the attendance attests, is a real ‘‘hot button.’’ The organizers (and the program chair, Lori Grob, in particular) deserve kudos for a job well done.